This is comment list on form factors available in the Irena package

Content

1. Form factors parameters
2. List of form factors
    2.1. Spheroid
    2.2. Integrated Spheroid
    2.3. Cylinder & cylinderAR
    2.4. Algebraic_Globules
    2.5. Algebraic_Rods
    2.6. Algebraic_Disks
    2.7. Unified_Sphere
    2.8. Unified_Rod & Unified_RodAR
    2.9. Unified_Disk
    2.10. CoreShell
    2.11. Tube = core shell cylinder
    2.12. Fractal Aggregate
3. Testing and using Form factors in users own code

Form Factor	Parameter(s)
spheroid	**AspectRatio = ParticlePar1**
Integrated_Spheroid	**AspectRatio=ParticlePar1**
Cylinders	**Length=ParticlePar1**
CylindersAR	**AspectRatio=ParticlePar1**
Unified_Disc	**thickness = ParticlePar1**
Unified_Rod	**length = ParticlePar1**
Unified_RodAR	**AspectRatio = ParticlePar1**
Unified_Sphere	**none needed**
Algebraic_Globules	**AspectRatio = ParticlePar1**
Algebraic_Rods	**AspectRatio = ParticlePar1, AR > 10**
Algebraic_Disks	**AspectRatio = ParticlePar1, AR < 0.1**
User	**uses user provided functions.** **There are two user provided functions necessary -** **F(q,R,par1,par2,par3,par4,par5)** **and V(R,par1,par2,par3,par4,par5)** **the names for these need to be provided in strings...** **the input is q and R in angstroms**
Tube	**length=ParticlePar1** **//length in A** **WallThickness=ParticlePar2 //in A** **CoreRho =ParticlePar3** **//rho [10^10 cm-2] (not delta rho squared!!!) of core material** **ShellRho =ParticlePar4** **//rho [10^10 cm-2] (not delta rho squared!!!) of shell material** **SolventRho =ParticlePar5** **//rho [10^10 cm-2] (not delta rho squared!!!) of surrounding medium (air=0)**
CoreShell	**CoreShellThickness=ParticlePar1**

//skin thickness in Angstroms
CoreRho =ParticlePar2
//rho [10^10 cm-2] (not delta rho squared!!!) of core material
ShellRho =ParticlePar3
//rho [10^10 cm-2] (not delta rho squared!!!) of shell material
SolventRho =ParticlePar4
//rho [10^10 cm-2] (not delta rho squared!!!) of surrounding medium (air=0)

Fractal aggregate **FractalRadiusOfPriPart=ParticlePar1**
//radius of primary particle
FractalDimension=ParticlePar2
//Fractal dimension

<u>List and graphs of each form factor included</u>

2.1. Spheroid

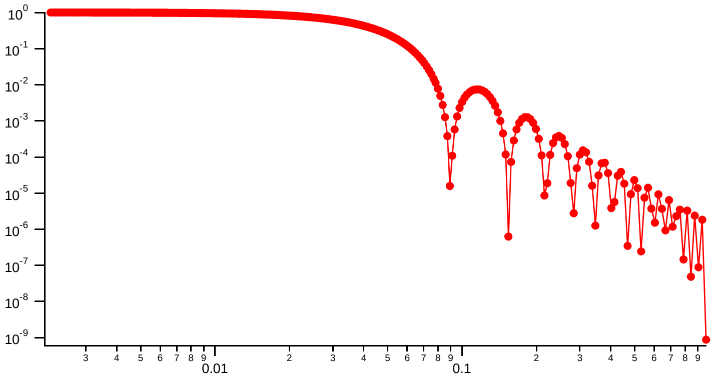
uses <u>sphere form factor</u> for aspect ratio between 0.99 and 1.01:

F^2 = 3/(QR^3))*(sin(QR)-(QR*cos(QR))

volume : V=((4/3)*pi*radius^3)

This calculation approximates integral ove R as rectangle (compare with Integrated spheroid).

graph for R = 50A



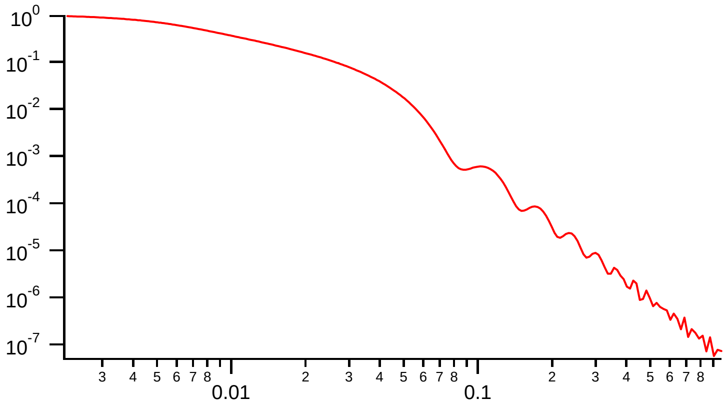
For aspect rations smaller than 0.99 and larger than 1.01 uses standard form factor for spheroid:

F = Integral of (3/(QR^3))*(sin(QR)-(QR*cos(QR)))

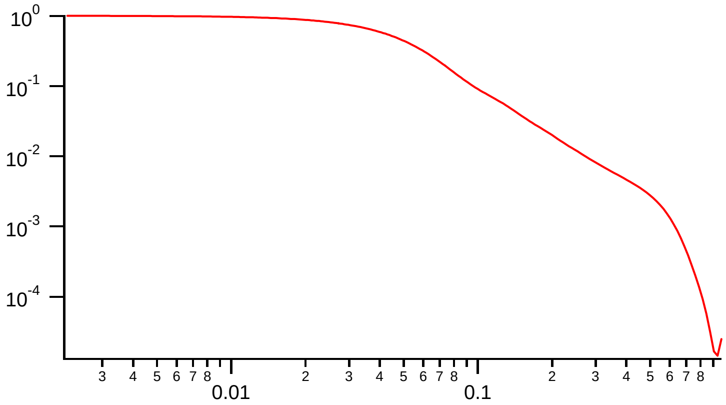
where QR=Qvalue*radius*sqrt(1+(((AR^2)-1)*CosTh^2))

over of CosTh = 0 to 1. This is numerically calculated using 50 points (step in CosTh = 0.02). Following graphs are examples:

AR = 10



AR=0.1



****
****
****

2.2. Integrated Spheroid

same code as in the spheroid, but in this case the code integrates over the width of the R bin.
Note, the bin star and end points are calcualted linearly (even for log-binned data) as half way distance:

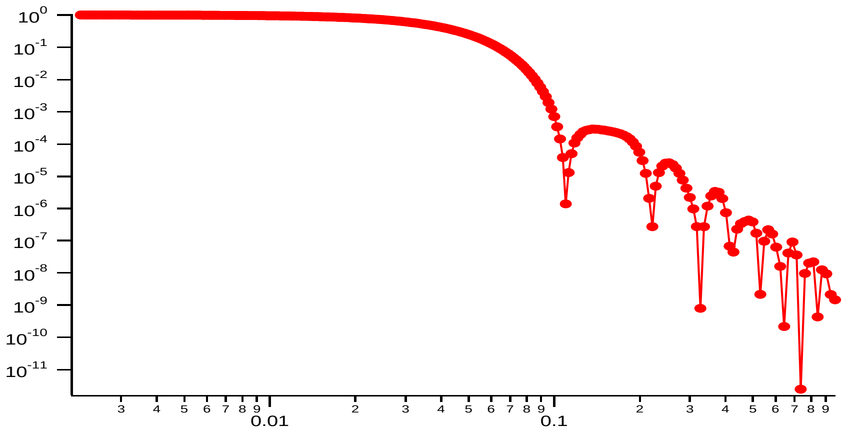
$R_{start} = (R_n + R_{n-1})/2$

$R_{end} = (R_n + R_{n+1})/2$

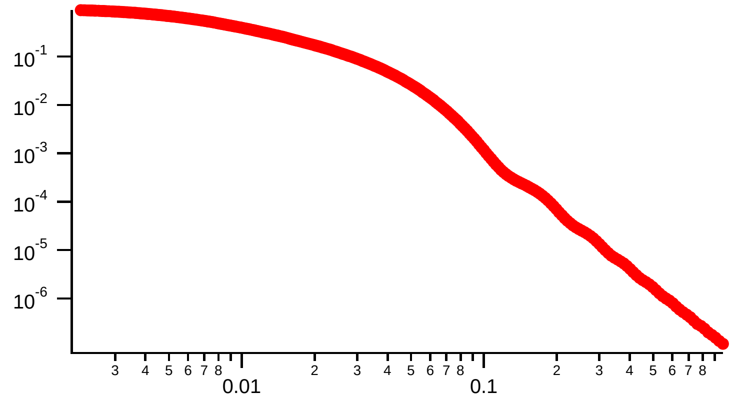
Uses adaptive steps to integrate essel function oscillations of the form factor over the width of the bin in R - note, the averaging is done including the volume of particles involved. This code is quite convoluted and time consuming. Its only reasonable use is for cases with wide bins in radius (R), when this removes some of the bessel function oscillations.

Examples with R width 40A, average size 50A (that means R varies from 30 to 70A). Note that the bessel function oscillations are somewho smooth out. With wider bins in R these oscillations may disappear all together.

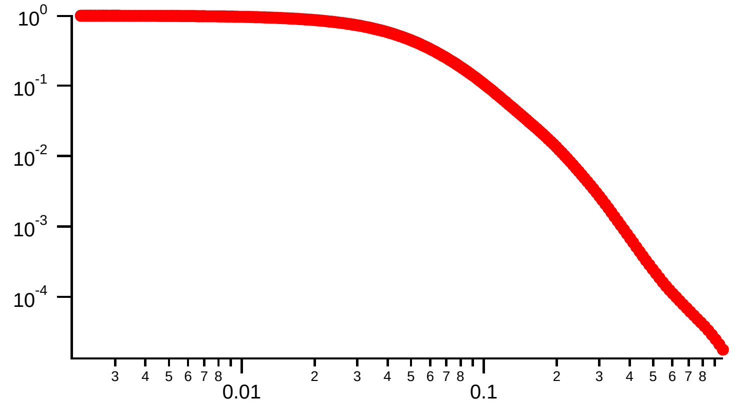
AR = 1 (sphere)



AR=10 (Spheroid)



AR=0.1 (spheroid)



****
****
****

2.3. Cylinder & cylinderAR

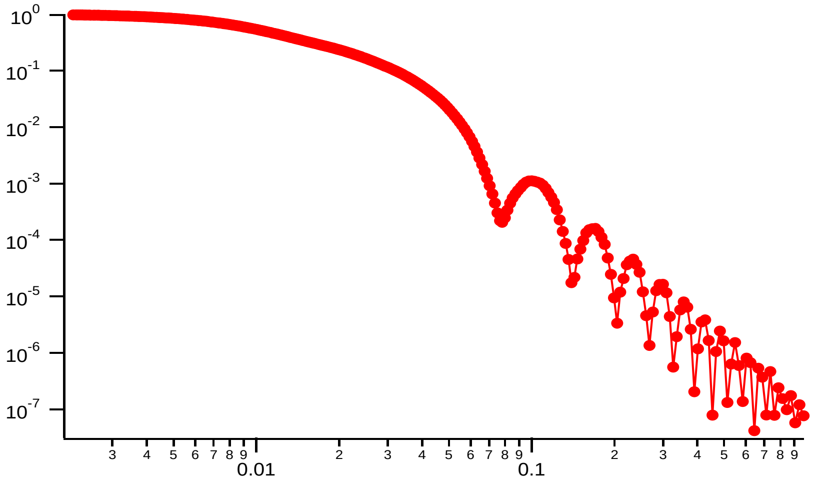
The code uses the following code to calculate form factor for cylinder. Note, that also this code is doing the same integration as integrated spheroid above (see 2).

Form factor = integral over (Ft) for Alpha = 0 to pi/2, Ft is below:

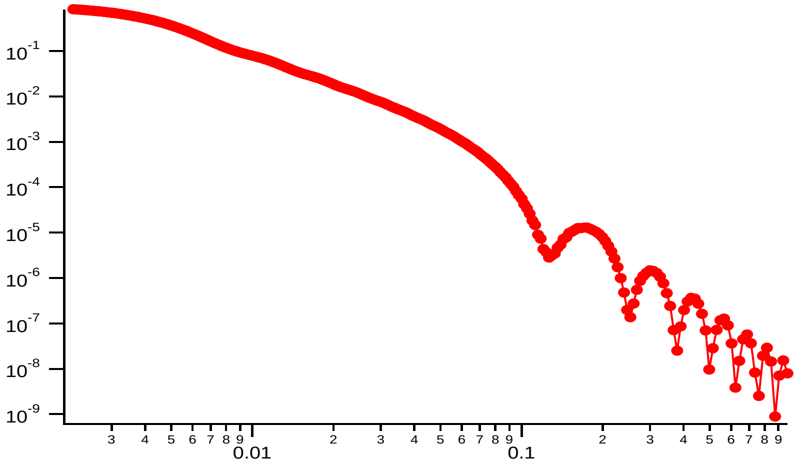
LargeBes=sin(0.5*Qvalue*length*Cos(Alpha)) / LargeBesArg
SmallBessDivided=BessJ(1, Qvalue*radius*Sin(Alpha))/Qvalue*radius*Sin(Alpha)
Ft = LargeBes*SmallBessDivided

Examples

Cylinder with length 500A and radius 50A.



Disk (cylinder) with radius 500A and length 50A.



****
****
****

2.4. Algebraic_Globules

Note, that for this model 1/3 <= Aspect ratio <=3 is appropriate

This is form factor created by Andrew Allen, it is a formula, which satisfies the basic form factor requirements:

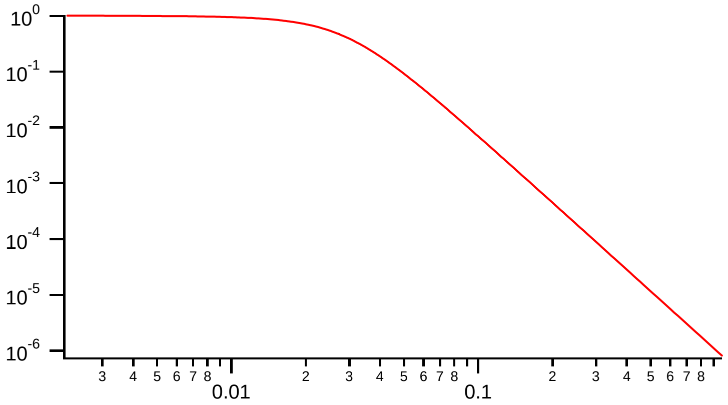
1. Terminate at 1 at small q
2. Place Gunier region in the right place
3. Have the right terminal slope

This is the code

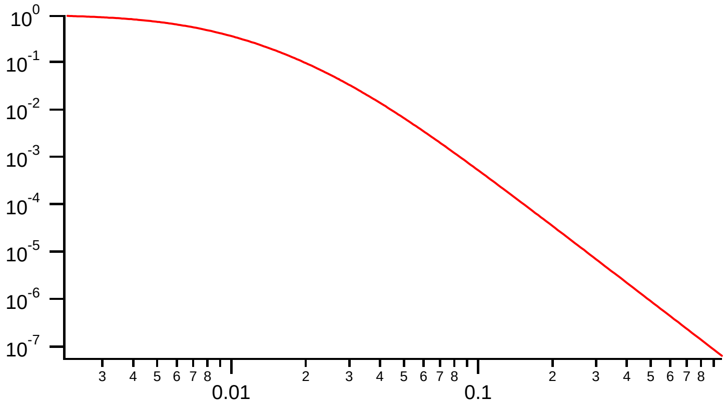
```
    if(AspectRatio<0.99)
        sqqt = sqrt(1-AspectRatio^2)
        argument = (2 - AspectRatio^2 + 2*sqqt)/(AspectRatio^2)
        surchi = (1 + AspectRatio^2 * ln(argument) / (2*sqqt)) / (2 * AspectRatio)
    elseif(AspectRatio>1.01)
        sqqt = sqrt(AspectRatio^2 - 1)
        argument = sqqt / AspectRatio
        surchi = (1 + AspectRatio^2 * asin(argument) / (sqqt)) / (2 * AspectRatio)
    else
        //AspectRatio==1
        surchi = 1
    endif
    QR = currentR * Q_vector
    bott = 1 + QR^2 * (2 + AspectRatio^2)/15 + 2 * AspectRatio * QR^4 / (9 * surchi)
    F^2 = 1  /  bott
```

Following are examples

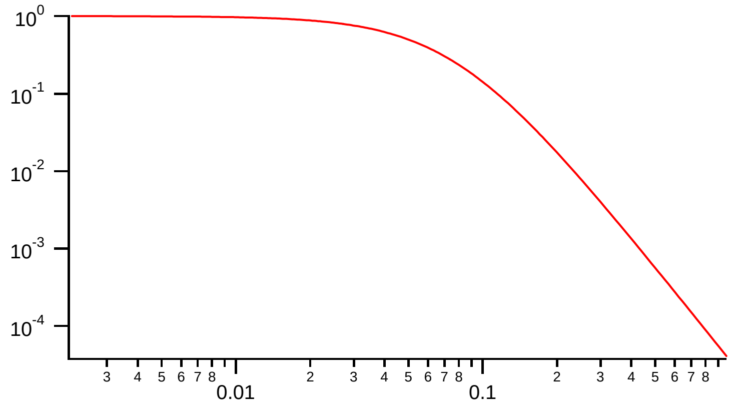
AR=1, R=50A



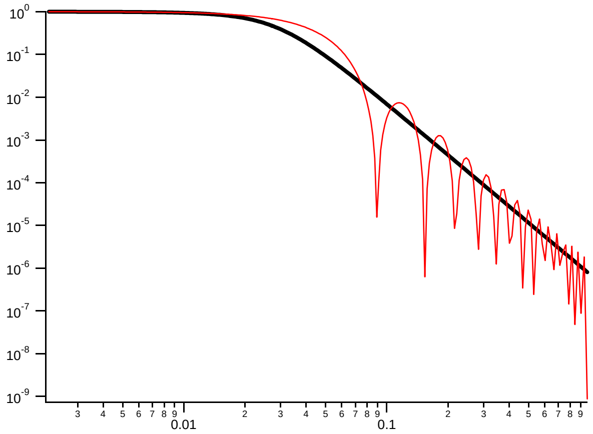
AR=10, R=50A



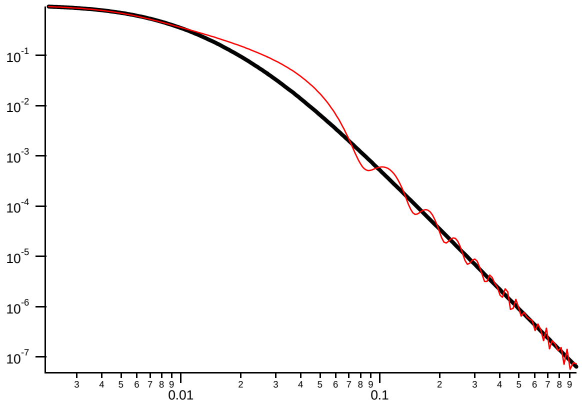
AR=0.1, R=50A



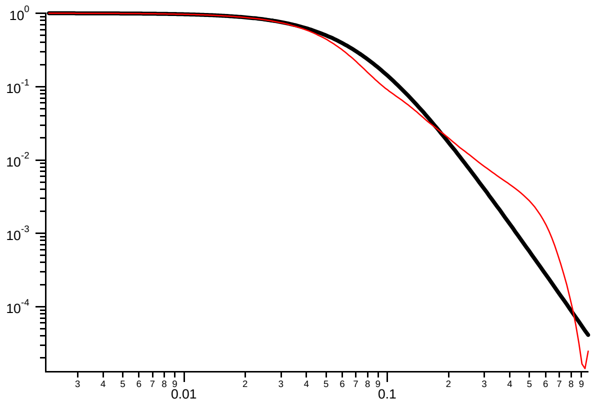
Comparison of Algebraic globule with spheroid:
AR=1



AR=10



AR=0.1



****
****
****

2.5. Algebraic_Rods

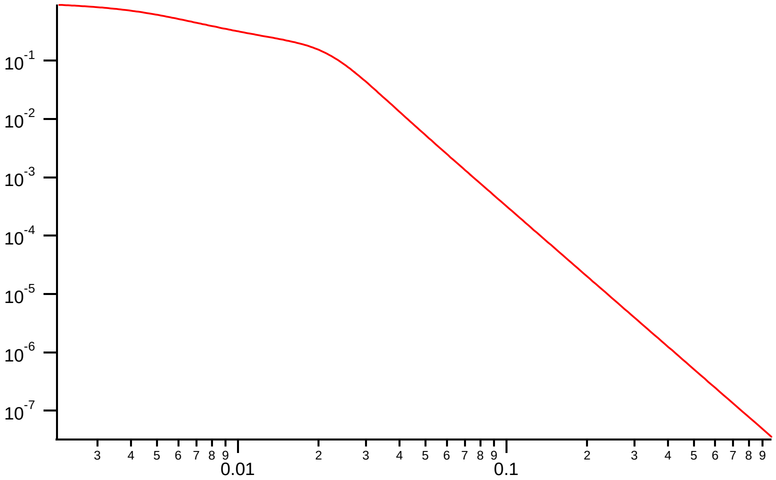
Note, that this formula is valid for Aspect ratio >=10

Another formula Andrew Allen...

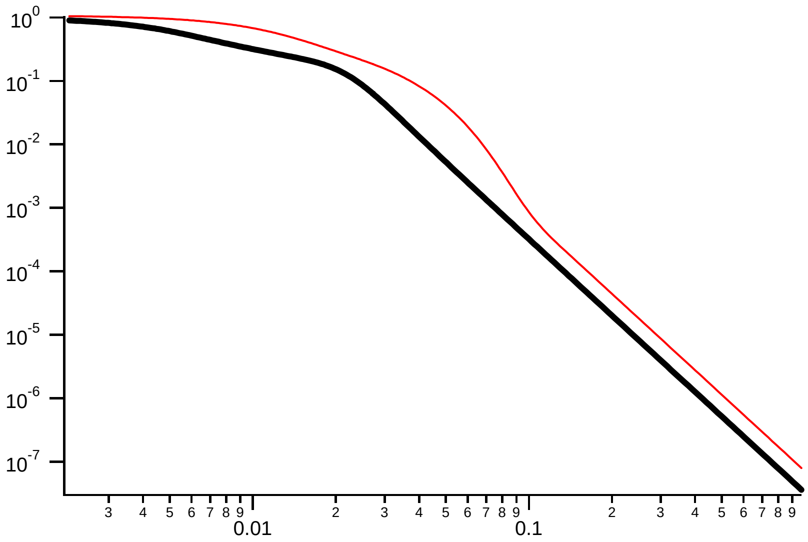
This is the code:

```
QR = R * Q
QH = Q * AspectRatio * R
topp = 1 + 2*pi*QH^3 * QR/(9 * (4 + QR^2)) + (QH^3 * QR^4)/8
bott = 1 + QH^2 * (1 + QH^2 * QR)/9 + (QH^4 * QR^7)/16
F^2 = topp/bott
```

Examples
AR10



Comparison with cylinder AR = 10



****
****
****

2.6. Algebraic_Disks

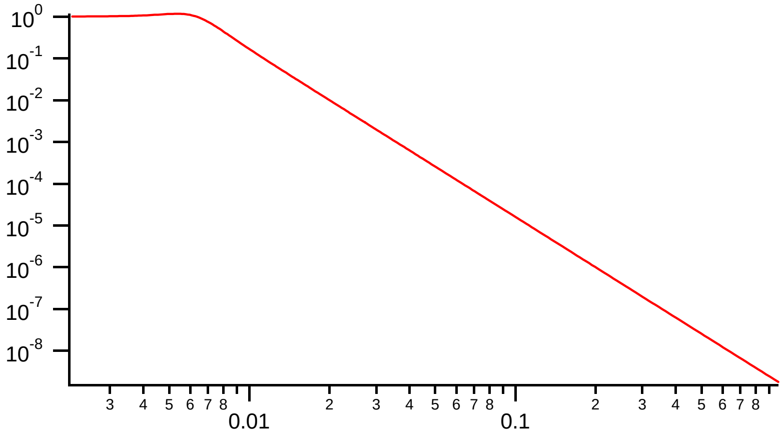
Note, that this formula is valid for Aspect ratio <=0.1

Another formula Andrew Allen...

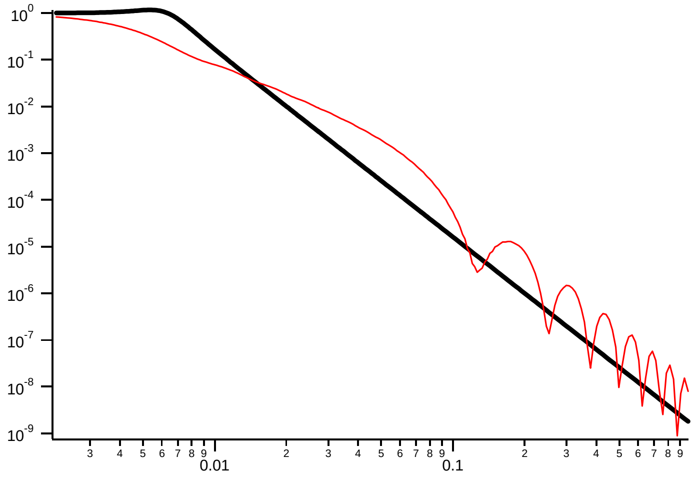
This is the code:

```
QH = Q * R
QRd = Q *  R * AspectRatio
topp = 1 + QRd^3 / (3 + QH^2) + (QH^2 * QRd / 3)^2
bott = 1 + QRd^2 * (1 + QH * QRd^2)/16 + (QH^3 * QRd^2 / 3)^2
F^2 = topp/bott
```

Examples
Radius 500A, AR = 0.1



Comparison with cylinder AR 0.1



****

****

****

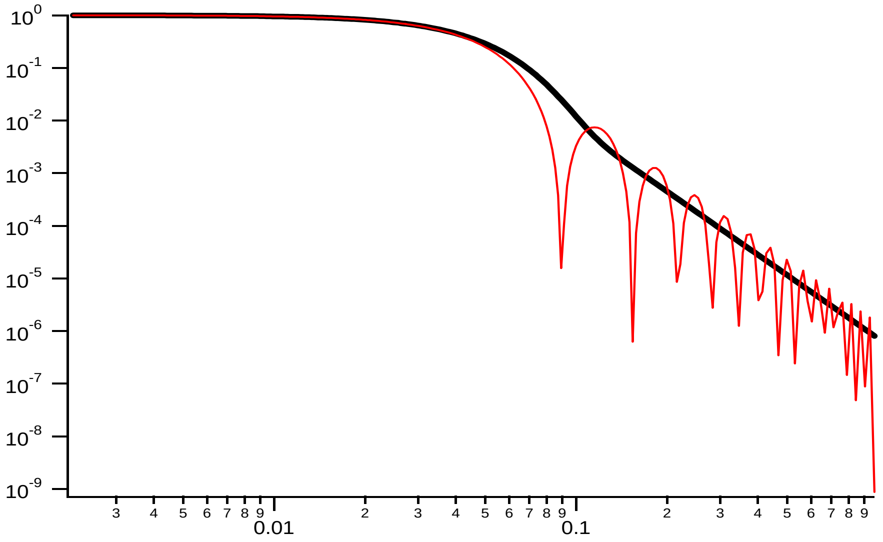
2.7. Unified_Sphere

This is formula from Unified fit model by Greg Beaucage (see Unified tool and documentation for it). The parameters are calculated from the code in the manual for each different shape. Specific formulas for these shapes were provided by Dale Schaefer...

This is the code:

```
G1=1
P1=4
Rg1=sqrt(3/5)*radius
B1=1.62*G1/Rg1^4
QstarVector=qvalue/(erf(qvalue*Rg1/sqrt(6)))^3
F^2 = G1*exp(-qvalue^2*Rg1^2/3)+(B1/QstarVector^P1)
```

Example for R=50A compared with the spheroid with aspect ratio =1



****
****
****

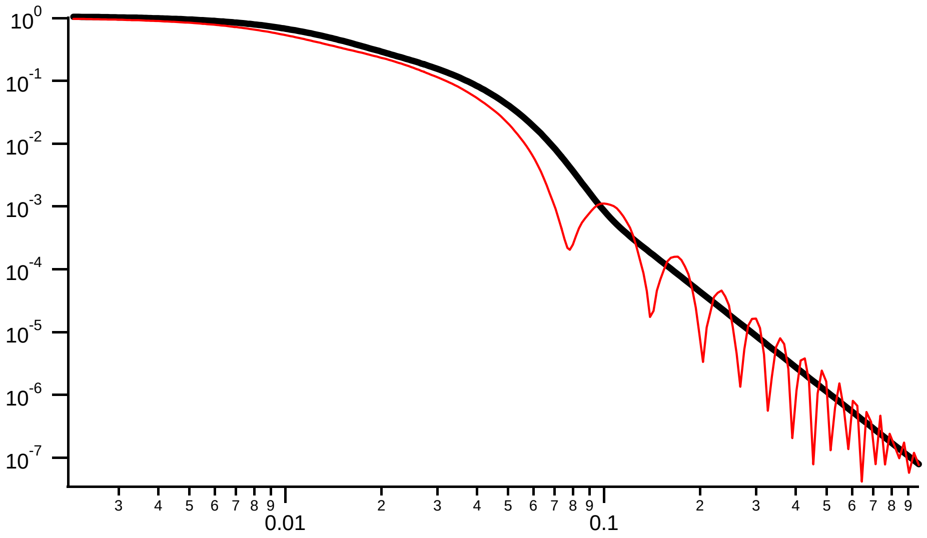
2.8. Unified_Rod & Unified_rodAR

This is formula from Unified fit model by Greg Beaucage (see Unified tool and documentation for it). The parameters are calculated from the code in the manual for each different shape. Specific formulas for these shapes were provided by Dale Schaefer...

This is the code:

```
G2 =1
Rg2=sqrt(Radius^2/2+Length^2/12)
B2=G2*pi/length
P2=1
Rg1=sqrt(3)*Radius/2
RgCO2=Rg1
G1=2*G2*Radius/(3*Length)
B1=4*G2*(Length+Radius)/(Radius^3*Length^2)
P1=4
QstarVector=qvalue/(erf(qvalue*Rg2/sqrt(6)))^3
A=G2*exp(-qvalue^2*Rg2^2/3)+(B2/QstarVector^P2) * exp(-RGCO2^2 * qvalue^2/3)
QstarVector=qvalue/(erf(qvalue*Rg1/sqrt(6)))^3
F^2 = A + G1*exp(-qvalue^2*Rg1^2/3)+(B1/QstarVector^P1)
```

Example for R=50A and length 500A compared with the cylinder



****
****
****

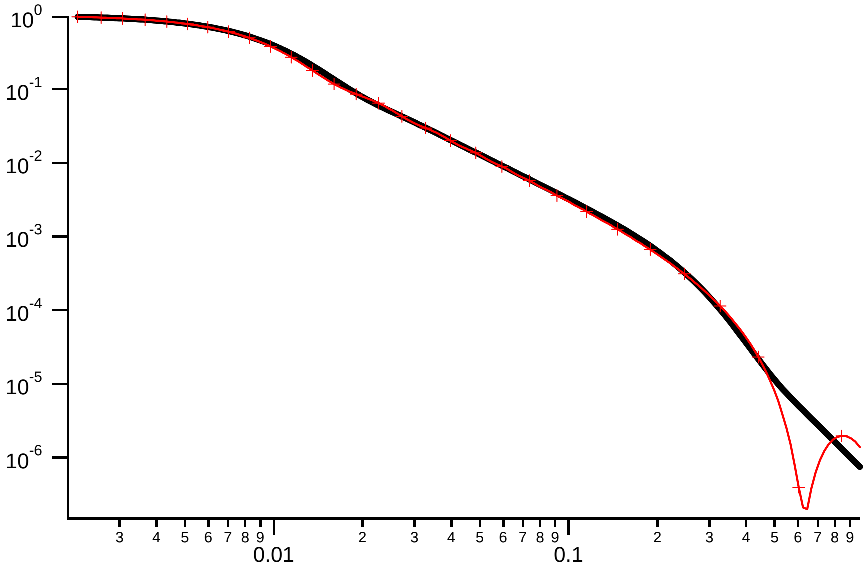
2.9. Unified_Disk

This is formula from Unified fit model by Greg Beaucage (see Unified tool and documentation for it). The parameters are calculated from the code in the manual for each different shape. Specific formulas for these shapes were provided by Dale Schaefer...

This is the code:

```
        G2=1
        Rg2=sqrt(Radius^2/2+thickness^2/12)
        B2=G2*2/(radius^2)//dws guess
        P2=2
        Rg1=sqrt(3)*thickness/2// Kratky and glatter = Thickness/2
        RgCO2=1.1*Rg1
        G1=2*G2*thickness^2/(3*radius^2)
        B1=4*G2*(thickness+Radius)/(Radius^3*thickness^2)//same as rod
        P1=4
        QstarVector=Q/(erf(Q*Rg2/sqrt(6)))^3
        A=G2*exp(-Q^2*Rg2^2/3)+(B2/QstarVector^P2) * exp(-RGCO2^2 * Q^2/3)
        QstarVector=Q/(erf(Q*Rg1/sqrt(6)))^3
        F^2 = A + G1*exp(-Q^2*Rg1^2/3)+(B1/QstarVector^P1)
```

Example for R=250A and thickness 10A compared with the cylinder



****
****
****

2.10. CoreShell

Note, this form factor calculation also includes integration over the width of bin in radii (same as integrated spheroid and cylinder).

Note: Input contrast is delta-rho-squared of core to surrounding (solvent/matrix)! this is very important to keep in mind.

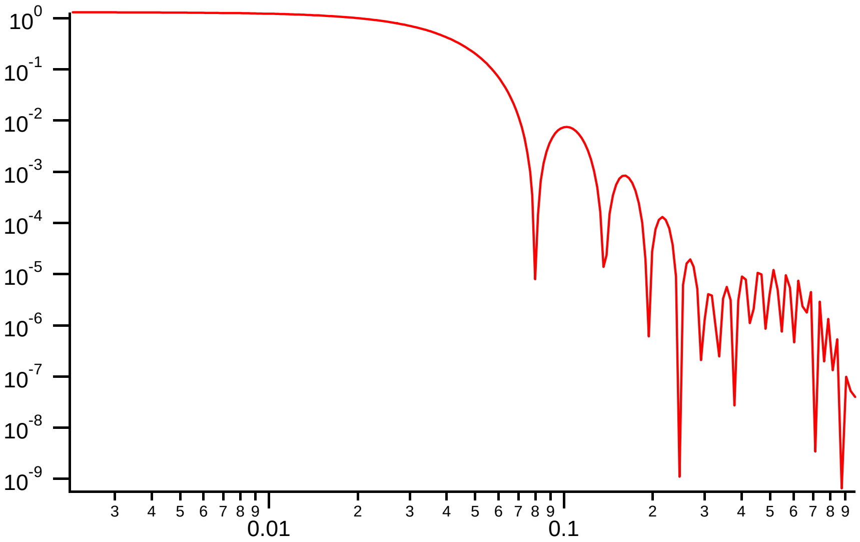
Code (heavily simplified!):

```
        Contrast = CoreRho - ShellRho
        //core
        Result1=(3/(Q*R)^3)*(sin(Q*R)-(Q*R*cos(Q*R)))^2  * Contrast * (4/3 * pi * R^3)
```

```
          //Now add the shell (skin) , thickness Rshell
          r = R+Rshell
          Contrast = ShellRho - SolventRho
          Result2 = (3/(Q*r)^3)*(sin(Q*r)-(Q*r*cos(Q*r)))^2 * Contrast * (4/3 * pi * r^3)

          //summ them together and normalize by the total particle volume
          F^2 =( result1 + result2 )^2 / (4/3 * pi * r^3)
```

Note, that to my surprise these calculations (copied from NIST Form factors) do not normalize correctly to 1 at low q. The reason is that the weighting is done by volume and contrast. I'll need to look into this again and in detail...

Example, Radius 50A, skin thickness 10A, contrast ratio 0.6



****
****
****

2.11. Tube = core shell cylinder

Note, this form factor calculation also includes integration over the width of bin in radii (same as integrated spheroid and cylinder).

This code has been developed some time ago and I am not sure about it's function...

Code (heavily simplified):

```
          Form factor is calculated for 500 points in Alpha for Alpha = 0 to pi/2
**************
          LargeBesArg=0.5*Qvalue*length*Cos(Alpha)
          if(LargeBesArg<1e-6)
                    LargeBes=1
          else
                    LargeBes=sin(LargeBesArg)/(LargeBesArg)
          endif

          SmallBesArg=Qvalue*radius*Sin(Alpha)
```

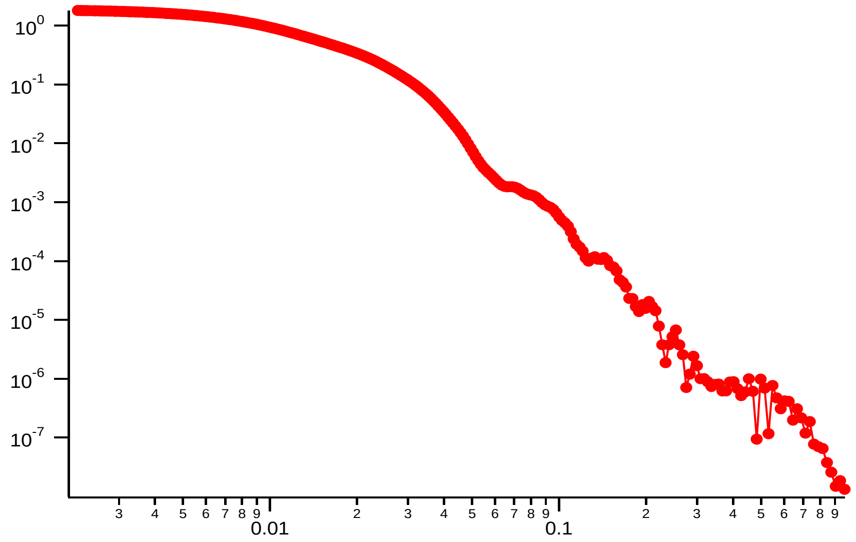
```
if (SmallBesArg<1e-10)
        SmallBessDivided=0.5
else
        SmallBessDivided=BessJ(1, SmallBesArg)/SmallBesArg
endif

LargeBesShellArg=0.5*Qvalue*(length+WallThickness)*Cos(Alpha)
if(LargeBesShellArg<1e-6)
        LargeBesShell=1
else
        LargeBesShell=sin(LargeBesShellArg)/(LargeBesShellArg)
endif

SmallBesShellArg=Qvalue*(radius+WallThickness)*Sin(Alpha)
if (SmallBesShellArg<1e-10)
        SmallBessShellDivided=0.5
else
        SmallBessShellDivided=BessJ(1, SmallBesShellArg)/SmallBesShellArg
endif
ratioOfVolumes=IR1T_TubeVolume(radius,Length) /
IR1T_TubeVolume(radius+WallThickness,Length)
```

F(Alpha) = 2 * ratioOfVolumes * (RhoCore - RhoShell) * (LargeBes * SmallBessDivided) + 2 * (RhoShell - RhoSlovent) * (LargeBesShell * SmallBessShellDivided)

****

F^2 = (integral over Alpha (from 0 to 1) over ((F^2) * sin(Alpha)))

Example
Length = 500A, Radius = 50A, Wall thickness=10A, Core contrast ratio =0.5



****
****
****

2.12. Fractal Aggregate

This form factor was requested by Dale Schaefer and I cannot very well guarantee its functionality....

code:

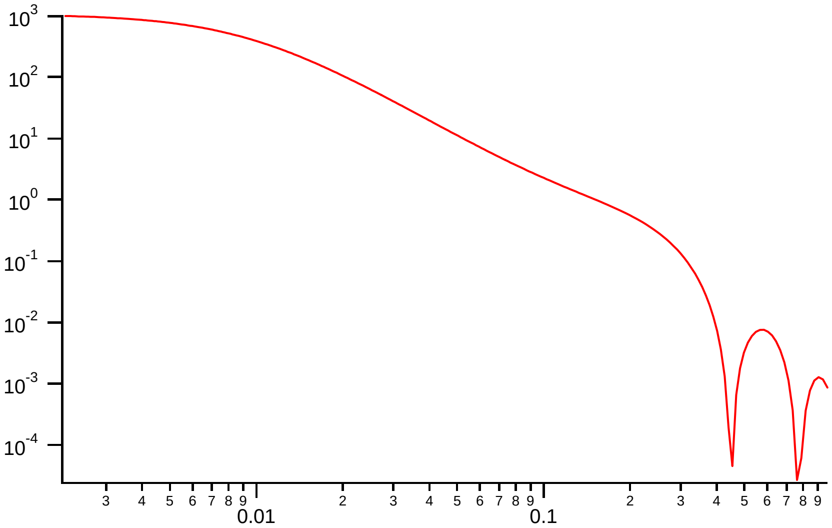
```
        f = IR1T_CalcSphereFormFactor(Qw[p],(2*Param1))
        //calculates the F(Q,r) part fo formula
        //this is same as for sphere of diameter = 2*Param1
        //(= radius of primary particle, which is hard sphere)
        //fractal part is next
        F^2 =f^2 * IR1T_CalculateFractAggSQPoints(Qw[p],currentR,Param1, Param2)
```

where

```
IR1T_CalculateFractAggSQPoints(Qvalue,R,r0, D) is
          QR=Qvalue*R
          part1=1
          part2=(qR*r0/R)^-D
          part3=D*(exp(gammln(D-1)))
          part5= (1+(qR)^-2)^((D-1)/2)
          part4=abs(sin((D-1)*atan(qR)))
           return (part1+part2*part3*part4/part5)
```

Note, that parameters are :
Param1 - radius of primary particle
param2 - fractal dimension of the fractal particles

Example for R=100A, radius of primary particle 10 A and fractal dimension 2.5.



Comment: Note, that this is not scaled correctly at all... I have no idea why - apparently this formula is either wrongly coded or plainly does not behave right.

3. Testing and using Form factors in users own code

To verify that the form factor works for you and to use the form factor if your own functions use following process and functions:

1. Generate Q wave with Qs for which the data are to be calculated

2. Generate intensity wave (will be redimesnioned as necessary, so the only thing is, it should be double precision).

3. Generate distributipon of radii wave - if you want to use single R, create wave with single point

4. decide what you want to calculate:

F^2	powerFct=0
V*F^2	powerFct=1
V^2 * F^2	powerFct=2

5. Run following command:

```
IR1T_GenerateGMatrix(R_FF,Q_wave,R_dist,powerFct,"form factor
       name",param1,param2,param3,param4,param5, "", "")
```

This function will return R_intensity, which is generally matrix with dimensions numpoints(Q_vector) x numpoints(R_dist), if R_dist has 1 point only, returned is wave (vector) as expected and reasonable...

The param1 - param5 are form factor parameters, as desribed in chapter 1, the "" at the end are for user form factor functions (there go the strings with names of user form factor and volume function).

"form factor name" is name from list in chapter 1.

6. Create log-log plot of the data if R_dist has single point. If R_dist has more point, well, you have to pull out the right column of data you need to plot.

Note, that if the IR1T_GenerateGMatrix function returns wave of NaN values if unknown name of form factor is passed in.

Example of code:

```
make/N=100 Q_wave
Q_wave=0.001+p/100
       //will create 100 points wave with values 0.001 to 1) values
Make/O/D R_FF
       //makes some place for form factor
make R_dist
R_dist=50
//or
//make/N=3 R_dist
//R_dist={10,50,100}
       //creates R distribution and sets values
IR1T_GenerateGMatrix(R_FF,Q_wave,R_dist,powerFct,"form factor
       name",param1,param2,param3,param4,param5, "", "")
       //Note, above lines belong on one line together!
       // replace powerFct with 0, 1,or 2!
// replace "form factor name" with name of form factor you want to use
Display R_FF vs Q_wave
ModifyGraph log=1
      //creates log-log graph of
```